**Министерство науки и высшего образования РФ**

**Государственное образовательное учреждение высшего**

**профессионального образования**

**Белгородский Государственный Технологический Университет им**

**В. Г. Шухова**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем.

Расчётно-графическое задание по дисциплине

«Технологии web-программирования»

Разработка мобильного приложения «Guitar shop».

Выполнил:

студент группы ПВ-42

Овчинников М.В

Проверил:

Картамышев С.В.

Белгород 2021 г.

**Цель работы.**

Целью работы является разработка REST API приложения по поиску покупки музыкальных инструментов. Пользователь может смотреть новости в мире музыки, покупать музыкальные инструменты, находить места где продаются или куда доставляются музыкальные инструменты. Но все данные функции доступны только зарегистрированным пользователям. Пользователь имеет некоторое количество баллов которые может потратить на покупку через приложение. Новости при нажатии открываются в браузере который выберет пользователь. Так же пользователь имеет свой профиль в котором отображается информация о нем.

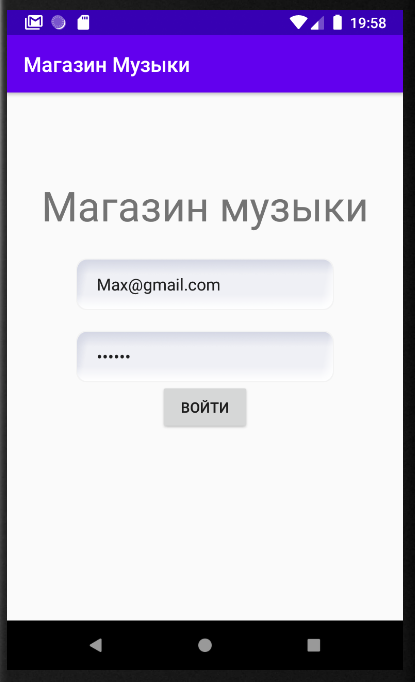
Для реализации задачи разработка была разбита две составляющих:

1. Мобильное приложение – приложение, использующееся на стороне клиента, с которого будут делаться определённые запросы на сервер. Для запросов к серверу использовалась библиотека Retrofit.
2. Бекенд – приложение, которое будет принимать запросы с клиента и обрабатывать их. Для реализации использовался язык программирования Kotlin вместе с фреймворком Spring. А в качестве СУБД выбор был сделан в пользу Postgresql.

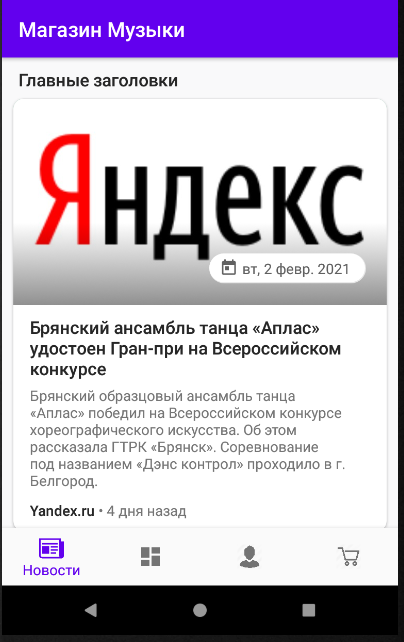
**Ход работы.**

**Внешний вид приложения.**

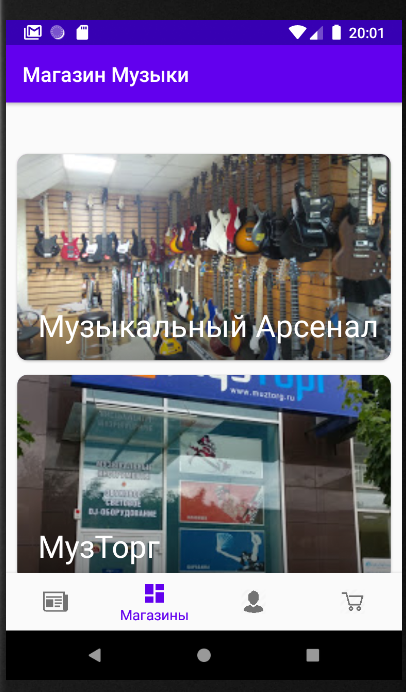
Страница авторизации.



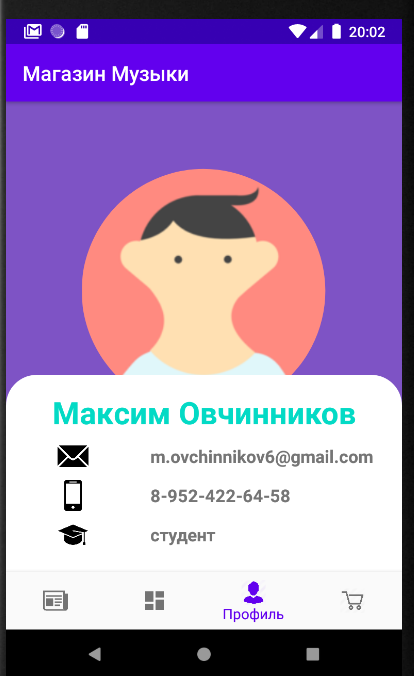
Страница новостей.



Страница магазинов.



Страница пользователя.



**Активити Андроид.**

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/container"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<androidx.fragment.app.FragmentContainerView

android:id="@+id/nav\_host\_fragment"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:layout\_above="@+id/nav\_view" />

<com.google.android.material.bottomnavigation.BottomNavigationView

android:id="@+id/nav\_view"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:layout\_marginStart="0dp"

android:layout\_marginEnd="0dp"

android:background="?android:attr/windowBackground"

android:layout\_alignParentBottom="true"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintLeft\_toLeftOf="parent"

app:layout\_constraintRight\_toRightOf="parent"

app:menu="@menu/bottom\_nav\_menu"/>

</RelativeLayout>

<?xml version="1.0" encoding="utf-8"?>

<layout>

<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

xmlns:app="http://schemas.android.com/apk/res-auto">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Магазин музыки"

android:textSize="40sp"

android:layout\_marginBottom="25dp"

app:layout\_constraintBottom\_toTopOf="@id/login"

app:layout\_constraintStart\_toStartOf="@id/login"

app:layout\_constraintEnd\_toEndOf="@id/login"/>

<EditText

android:id="@+id/login"

android:layout\_width="250dp"

android:layout\_height="50dp"

android:hint="Login"

android:text="Max@gmail.com"

android:inputType="text"

android:paddingStart="20dp"

android:layout\_marginBottom="140dp"

android:textSize="16sp"

android:background="@drawable/edit\_text\_background"

app:layout\_constraintTop\_toTopOf="parent"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"/>

<EditText

android:id="@+id/password"

android:layout\_width="250dp"

android:layout\_height="50dp"

android:hint="Password"

android:text="max123"

android:inputType="textPassword"

android:paddingStart="20dp"

android:textSize="16sp"

android:background="@drawable/edit\_text\_background"

app:layout\_constraintTop\_toBottomOf="@id/login"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

android:layout\_marginTop="20dp"

/>

<Button

android:id="@+id/enter"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Войти"

app:layout\_constraintTop\_toBottomOf="@id/password"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"/>

</androidx.constraintlayout.widget.ConstraintLayout>

</layout>

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".ui.market.MarketFragment" >

<androidx.core.widget.NestedScrollView

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical">

<LinearLayout

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Баллы: "

android:textSize="20dp" />

<TextView

android:id="@+id/text\_market"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="10000"

android:textSize="20dp" />

</LinearLayout>

<androidx.recyclerview.widget.RecyclerView

android:id="@+id/market\_recycler\_view"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:orientation="vertical"

android:scrollbars="vertical"

app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"></androidx.recyclerview.widget.RecyclerView>

</LinearLayout>

</androidx.core.widget.NestedScrollView>

</androidx.constraintlayout.widget.ConstraintLayout>

<?xml version="1.0" encoding="utf-8"?>

<androidx.constraintlayout.widget.ConstraintLayout

xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:app="http://schemas.android.com/apk/res-auto"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/profiler\_constraint\_layout"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:background="@color/colorPurpleL"

tools:layout\_editor\_absoluteY="81dp">

<ImageView

android:id="@+id/profile\_pic"

android:layout\_width="240dp"

android:layout\_height="240dp"

android:layout\_marginStart="8dp"

android:layout\_marginTop="64dp"

android:layout\_marginEnd="8dp"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="parent"

app:srcCompat="@drawable/purple\_frame"

android:background="@drawable/avatar\_1\_raster"

tools:background="@tools:sample/avatars"

tools:src="@drawable/purple\_frame" />

<View

android:id="@+id/user\_data\_card"

android:layout\_width="0dp"

android:layout\_height="191dp"

android:layout\_marginTop="64dp"

android:background="@drawable/rounded\_rect"

app:layout\_constraintBottom\_toBottomOf="parent"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/profile\_pic"

app:layout\_constraintVertical\_bias="1.0"/>

<TextView

android:id="@+id/profile\_user\_name"

android:gravity="center\_horizontal"

android:layout\_width="0dp"

android:layout\_height="40dp"

android:layout\_marginStart="8dp"

android:layout\_marginTop="16dp"

android:layout\_marginEnd="8dp"

android:text="Максим Овчинников"

android:textAlignment="center"

android:textColor="@color/colorAccent"

android:textSize="30sp"

android:textStyle="bold"

app:layout\_constraintEnd\_toEndOf="parent"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toTopOf="@+id/user\_data\_card" />

<ImageView

android:id="@+id/mail"

android:layout\_width="30dp"

android:layout\_height="30dp"

android:layout\_marginTop="8dp"

android:layout\_marginStart="50dp"

app:layout\_constraintStart\_toStartOf="parent"

app:layout\_constraintTop\_toBottomOf="@+id/profile\_user\_name"

app:layout\_constraintEnd\_toStartOf="@id/mail\_text"

android:background="@drawable/mail"/>

<TextView

android:id="@+id/mail\_text"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="m.ovchinnikov6@gmail.com"

android:textSize="17sp"

android:textStyle="bold"

android:layout\_marginStart="60dp"

app:layout\_constraintTop\_toTopOf="@+id/mail"

app:layout\_constraintStart\_toEndOf="@id/mail"

app:layout\_constraintBottom\_toBottomOf="@id/mail"

/>

<ImageView

android:id="@+id/phone"

android:layout\_width="30dp"

android:layout\_height="30dp"

android:layout\_marginTop="8dp"

app:layout\_constraintStart\_toStartOf="@id/mail"

app:layout\_constraintTop\_toBottomOf="@+id/mail"

app:layout\_constraintEnd\_toEndOf="@id/mail"

android:background="@drawable/cell\_phone"/>

<TextView

android:id="@+id/phone\_text"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="8-952-422-64-58"

android:textSize="17sp"

android:textStyle="bold"

app:layout\_constraintTop\_toTopOf="@+id/phone"

app:layout\_constraintStart\_toStartOf="@id/mail\_text"

app:layout\_constraintBottom\_toBottomOf="@id/phone"

/>

<ImageView

android:id="@+id/edu"

android:layout\_width="30dp"

android:layout\_height="30dp"

android:layout\_marginTop="8dp"

app:layout\_constraintStart\_toStartOf="@id/phone"

app:layout\_constraintTop\_toBottomOf="@+id/phone"

app:layout\_constraintEnd\_toEndOf="@id/phone"

android:background="@drawable/grad"/>

<TextView

android:id="@+id/edu\_text"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="студент"

android:textSize="17sp"

android:textStyle="bold"

app:layout\_constraintTop\_toTopOf="@+id/edu"

app:layout\_constraintStart\_toStartOf="@id/phone\_text"

app:layout\_constraintBottom\_toBottomOf="@id/edu"

/>

</androidx.constraintlayout.widget.ConstraintLayout>

Бекенд.

Схема базы данных.

Сущности ORM.

@Entity(name = "authority")

@FieldDefaults(level = AccessLevel.PRIVATE)

public class Authority implements GrantedAuthority, Serializable {

@Id

@Getter

@Setter

@GeneratedValue(strategy = GenerationType.IDENTITY)

Long id;

String name;

@Override

public String getAuthority() {

return name;

}

}

@Entity(name = "guitar")

@Data

@FieldDefaults(level = AccessLevel.PRIVATE)

public class Guitar {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

Long id;

@NotNull

String name;

@NotNull

Integer amount;

@NotNull

Integer price;

@NotNull

String description;

@NotNull

Integer weight;

@NotNull

Integer length;

@NotNull

Integer height;

@NotNull

String url;

@ManyToOne(cascade = CascadeType.ALL)

@JoinColumn(name = "type\_id")

Type type;

}

@Entity(name = "type")

@Data

@FieldDefaults(level = AccessLevel.PRIVATE)

public class Type {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

Long id;

@NotNull

String name;

@OneToMany(mappedBy = "type")

Set<Guitar> guitars;

}

@Entity(name = "usr")

@Data

@FieldDefaults(level = AccessLevel.PRIVATE)

public class UserAccount implements UserDetails {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

Long id;

@NotNull

String username;

@NotNull

String password;

@NotNull

Integer cash;

boolean accountNonExpired;

boolean accountNonLocked;

boolean credentialsNonExpired;

boolean enabled;

@ManyToMany(fetch = FetchType.EAGER)

@JoinTable(name = "usr\_authorities",

joinColumns = @JoinColumn(name = "usr\_id", referencedColumnName = "id"),

inverseJoinColumns = @JoinColumn(name = "authority\_id", referencedColumnName = "id"))

Set<Authority> authorities;

@Override

public Collection<Authority> getAuthorities() {

return authorities;

}

@Override

public boolean isAccountNonExpired() {

return accountNonExpired;

}

@Override

public boolean isAccountNonLocked() {

return accountNonLocked;

}

@Override

public boolean isCredentialsNonExpired() {

return credentialsNonExpired;

}

@Override

public boolean isEnabled() {

return enabled;

}

}

@Service

@RequiredArgsConstructor

public class GuitarServiceImpl implements GuitarService {

private final GuitarRepository guitarRepository;

private final GuitarMapper guitarMapper;

private final GuitarMainPageMapper guitarMainPageMapper;

@Override

public List<GuitarMainPageResponse> getGuitars() {

return guitarMainPageMapper.toDTOs(guitarRepository.getAllBy());

}

@Override

public GuitarResponse getGuitarById(Long id) {

return guitarMapper.toDTO(guitarRepository.getGuitarById(id));

}

}

@Service

@RequiredArgsConstructor

public class UserAccountServiceImpl implements UserAccountService {

private final UserRepository userRepository;

private final GuitarRepository guitarRepository;

private final UserMapper userMapper;

@Override

@PreAuthorize("hasAuthority('ROLE\_USER')")

public Integer userBuyGuitars(Principal principal, Long id) {

UserAccount user = userRepository.getUserAccountByUsername(principal.getName());

try {

return userTryBuyGuitars(user, id);

}

catch (UserLowCashException | GuitarsLowAmountException e){

System.out.println(e.toString());

return -1;

}

}

@Override

public UserResponse getUserByUsername(String username) {

return userMapper.toDTO(userRepository.getUserAccountByUsername(username));

}

private Integer userTryBuyGuitars(UserAccount user, Long id) throws GuitarsLowAmountException, UserLowCashException {

Integer totalPrice = 0;

Guitar guitarFromDB = guitarRepository.getGuitarById(id);

if (guitarFromDB.getAmount() > 0) {

totalPrice += guitarFromDB.getPrice();

guitarFromDB.setAmount(guitarFromDB.getAmount() - 1);

} else throw new GuitarsLowAmountException("Guitar amount low");

if (totalPrice <= user.getCash()) {

user.setCash(user.getCash() - totalPrice);

userRepository.save(user);

guitarRepository.save(guitarFromDB);

return user.getCash();

}

else throw new UserLowCashException("User cash low");

}

@Override

public AuthorizationToken authenticationUser(RequestAuth requestAuth, String authorization) {

String uri = "http://localhost:8082/oauth/token";

RestTemplate restTemplate = new RestTemplate();

restTemplate.getMessageConverters().add(new ObjectToUrlEncodedConverter(new ObjectMapper()));

HttpEntity<Map<String, Object>> request =

new HttpEntity<>(createBody(requestAuth), createHeaders(authorization));

return restTemplate.postForObject(uri, request , AuthorizationToken.class);

}

private HttpHeaders createHeaders(String authorization){

HttpHeaders headers = new HttpHeaders();

headers.setBasicAuth(authorization.split(" ")[1]);

headers.setContentType(MediaType.APPLICATION\_FORM\_URLENCODED);

return headers;

}

private Map<String, Object> createBody(RequestAuth requestAuth){

Map<String, Object> bodyRequest = new HashMap<>();

bodyRequest.put("grant\_type", requestAuth.getGrant\_type());

bodyRequest.put("username", requestAuth.getUsername());

bodyRequest.put("password", requestAuth.getPassword());

return bodyRequest;

}

}

@Service

@RequiredArgsConstructor

public class UserDetailsServiceImpl implements UserDetailsService {

private final UserRepository userRepository;

@Override

public UserDetails loadUserByUsername(String s) throws UsernameNotFoundException {

return userRepository.getUserAccountByUsername(s);

}

}

@Configuration

@EnableAuthorizationServer

@RequiredArgsConstructor

@EnableGlobalMethodSecurity(prePostEnabled = true)

public class AuthorizationServerConfiguration extends AuthorizationServerConfigurerAdapter {

private final DataSource dataSource;

private final AuthenticationManager authenticationManager;

private final UserDetailsService userDetailsService;

private final PasswordEncoder passwordEncoder;

@Bean

public TokenStore tokenStore(){

return new JdbcTokenStore(dataSource);

}

@Bean

public OAuth2AccessDeniedHandler oAuth2AccessDeniedHandler(){

return new OAuth2AccessDeniedHandler();

}

@Override

public void configure(ClientDetailsServiceConfigurer clients) throws Exception {

clients.jdbc(dataSource);

}

@Override

public void configure(AuthorizationServerSecurityConfigurer security) throws Exception {

security

.tokenKeyAccess("permitAll()")

.checkTokenAccess("isAuthenticated()")

.passwordEncoder(passwordEncoder);

}

@Override

public void configure(AuthorizationServerEndpointsConfigurer endpoints) throws Exception {

endpoints

.tokenStore(tokenStore())

.authenticationManager(authenticationManager)

.userDetailsService(userDetailsService);

}

}

@Configuration

public class MvcConfig implements WebMvcConfigurer {

@Override

public void addCorsMappings(CorsRegistry registry){

registry.addMapping("/\*\*")

.allowCredentials(true)

.allowedHeaders("Authorization", "Cache-Control", "Content-Type", "Accept", "Origin")

.allowedMethods("HEAD", "OPTIONS", "GET", "POST", "PUT", "DELETE", "PATCH")

.allowedOrigins("\*");

}

}

@Configuration

@EnableSwagger2

class SwaggerConfiguration {

@Bean

Docket api() {

return new Docket(DocumentationType.SWAGGER\_2)

.select()

.apis(RequestHandlerSelectors.any())

.paths(PathSelectors.any())

.build()

.securitySchemes(securitySchemes())

.securityContexts(securityContexts());

}

@Bean

List<? extends SecurityScheme> securitySchemes() {

return Collections.singletonList(

new ApiKey("Bearer", "Authorization", ApiKeyVehicle.HEADER.getValue())

);

}

@Bean

List<SecurityContext> securityContexts() {

return Collections.singletonList(

SecurityContext.builder()

.securityReferences(defaultAuth())

.forPaths(PathSelectors.regex("/.\*"))

.build()

);

}

@Bean

List<SecurityReference> defaultAuth() {

AuthorizationScope authorizationScope

= new AuthorizationScope("global", "accessEverything");

AuthorizationScope[] authorizationScopes = new AuthorizationScope[1];

authorizationScopes[0] = authorizationScope;

return Collections.singletonList(new SecurityReference("Bearer", authorizationScopes));

}

}

**Вывод.**

* результате выполнения данной работы получилось полноценное приложение с задуманным ранее функционалом. Началось всё с разработки бэкенда. Далее была спроектирована база данных, прописаны точки входа в сервер, и наконец проработана бизнес-логика в сервисах. И закончил созданием и версткой приложения.